

Administration et Sécurité des Réseaux : Administration Système

– M2102 –

Camille Coti

`camille.coti@lipn.univ-paris13.fr`

Département R&T, IUT de Villetaneuse, Université de Paris XIII



- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

Introduction du module

Organisation du module :

- 3 CM de 1h30 chacun
- 6 TP de 3H chacun

Ressources : slides, cours, polycopié

- Polycopié à lire pour préparer les TP
- TP1 : partie 2
- TP2 : parties 3 à 4.3
- TP3 : parties 4.4 à 6
- TP4 à 6 : parties 7 à 9.

Tout est en ligne : <http://lipn.fr/~coti/cours>

→ Module M2102

Plan du cours

- 1 Introduction du module
- 2 Administration système**
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

Machines administrées :

- Serveurs
- Parc de machines
- Équipements réseaux...

Utilisateurs : à distinguer des **administrateurs** !

- Ne font qu'utiliser la machine, ne l'administrent pas
- Peuvent faire des actions limitées. Exemple : pas installer de nouveau périphérique
- Niveau de connaissances variable : certains experts (ou se croyant comme tel...), d'autres ne savent utiliser que ce dont ils ont besoin

Administrateur système

Il assure des **tâches d'administration** sur la machine

→ Tout système nécessite d'avoir un administrateur

Tâches d'administration :

- Installation et mise à jour des systèmes
- Installation et mise à jour de logiciels
- Ajout et suppression d'utilisateurs
- Ajout et suppression de matériel, reconfiguration
- Sauvegardes et restaurations
- Surveillance du système
- Sécurité
- Monitoring
- Gestion de la documentation locale
- Aide aux utilisateurs
- ...

Administrer une machine ou un parc ?

Administrer **sa machine** :

- La maintenir à jour, appliquer des correctifs quand il en sort...
- Installer un logiciel quand on en a besoin
- Environnement relativement maîtrisé

Administrer **un parc de machines** :

- Hétérogène : machines différentes, équipements différents (routeurs, serveurs, stations de travail, photocopieuses, machines à café...)
- Complexité due à cette hétérogénéité, au nombre d'équipements...
- Notion de machines de production : besoin d'équipements et de logiciels fiables
- Environnement peu maîtrisé : utilisateurs, extérieur...

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs**
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

Plusieurs utilisateurs :

- Plusieurs utilisateurs peuvent utiliser une même machine
- Sur des serveurs : plusieurs utilisateurs ayant chacun des droits limités :
 - Une tâche ↔ un utilisateur qui n'a le droit que de l'exécuter
 - Exemple : identité sous laquelle tourne le serveur Apache. Serveur piraté → contention de la menace

Principe important : **ne donner à un utilisateur que les droits dont il a absolument besoin**

Réflexion en amont de la mise en place des comptes utilisateurs sur un système :

- Catégories d'utilisateurs? → Définition de *groupes*
- Utilisateurs réels (physiques) ou pour l'administration?
- Permissions qu'il est nécessaire d'accorder à ces utilisateurs?
- Politique des comptes : temporaires, permanents, etc...

Super-utilisateur

- Il a tous les droits !
- À utiliser avec parcimonie... doit être réservé aux tâches d'administration.
- Jamais utilisé en permanence !

Comptes utilisateurs

- Vrais utilisateurs, correspondent à une personne
- Attention à éviter d'avoir plusieurs personnes sur un même groupe
- Doivent avoir des permissions limitées

Comptes système

- Ne correspondent pas à des personnes réelles
- Utilisés par le système pour certaines tâches. Exemple : faire tourner un serveur
- Ne doivent avoir que les permissions dont ils ont besoin pour effectuer cette tâche en particulier
- Ne doivent pas pouvoir se connecter sur le système

Utilisateurs locaux

- N'existent que sur le système local
- Créés, configurés et maintenus sur le système local
- N'existent pas sur une autre machine du réseau

Utilisateurs réseau

- Gestion généralement centralisée sur un un service d'annuaire (NIS, LDAP, ActiveDirectory...)
- Sont connus de toutes les machines qui utilisent ce service d'annuaire

Création d'un utilisateur

Seul le super-utilisateur peut effectuer cette tâche

→ Raison évidente...

Commande `adduser`

- Crée un utilisateur
- Local ou réseau
- Avec ou sans mot de passe, avec ou sans répertoire personnel, etc...

Commande `useradd`

- Crée un utilisateur
- Uniquement local
- Même comportement que `adduser`, à part le point précédent

Mais ! Attention, le comportement de ces commandes dépend souvent de la version d'Unix et de la distribution utilisée. Lisez la doc en ligne locale !

Configuration de la création d'utilisateurs

Fichiers `/etc/login.defs` et `/etc/adduser.conf`

- `/etc/login.defs` permet de configurer `useradd`
- `/etc/adduser.conf` permet de configurer `adduser`

Exemple :

```
UID_MIN          1000
UID_MAX          60000
DEFAULT_HOME     yes
```

Profil par défaut : contenu du répertoire `/etc/skel`

- Squelette qui va être copié dans le répertoire personnel de chaque utilisateur créé
- Fichiers mais aussi répertoires

Obtenir des informations sur un utilisateur :

- Son iud, ses groupes : `id`

```
coti@maximum:~$ id coti
```

```
uid=3456(coti) gid=1000 groupes=1000
```

Obtenir des informations sur un utilisateur :

- Son iud, ses groupes : `id`

```
coti@maximum:~$ id coti
uid=3456(coti) gid=1000 groupes=1000
```

- Plus d'info : `finger`

```
coti@maximum:~$ finger figatellix
Login: figatellix                Name: Figatellix
Directory: /home/figatellix      Shell: /bin/bash
Last login Tue Feb  4 15:38 (CET) on tty1
New mail received Sun Apr 27 17:51 2014 (CEST)
      Unread since Mon Feb  3 16:07 2014 (CET)
No Plan.
```

Fichier /etc/passwd

- Contient les informations relatives aux comptes eux-mêmes
- Une ligne par compte
- 7 champs séparés par le symbole " : " :

Exemple :

```
figatellix:x:1000:1001:Figatellix,,,:/home/figatellix:/bin/bash
```

- 1 login
- 2 x, anciennement le mot de passe, non-affichable
- 3 user id, ou *uid* de l'utilisateur
- 4 numéro du groupe primaire (group id, ou *gid*) de l'utilisateur
- 5 nom complet de l'utilisateur et éventuelles remarques ;
- 6 chemin vers le répertoire personnel ;
- 7 shell de connexion

Fichier /etc/shadow

- Contient les informations relatives aux mots de passe des comptes, et les mots de passe (cryptés)
- Une ligne par compte
- 9 champs séparés par le symbole " : " :

Exemple :

```
figatellix:$6$ZzwWAP5z$DzF2Qf8pib3CY88EqeBlQz6KNC1:16104:0:99999:7:::
```

- 1 login
- 2 mot de passe, crypté
- 3 le nombre de jours entre le 1er janvier 1970 et la date de dernier changement du mot de passe
- 4 nombre de jours avant que le mot de passe puisse être changé
- 5 nombre de jours avant que le mot de passe doive être changé
- 6 nombre de jours durant lesquels l'utilisateur est prévenu de l'expiration de son mot de passe
- 7 nombre de jours entre l'expiration du mot de passe et la fermeture du compte
- 8 la date de fermeture du compte
- 9 champ réservé pour un éventuel usage ultérieur.

```
coti@maximum:~$ ls -l /etc/passwd /etc/shadow
-rw-r--r-- 1 root root 1762 févr. 14 14:51 /etc/passwd
-rw-r----- 1 root shadow 1196 févr. 14 14:51 /etc/shadow
```

Les deux appartiennent au super-utilisateur

- /etc/passwd est lisible par tout le monde
- /etc/shadow n'est lisible par le super-utilisateur et les membres du groupe shadow.
- modifiables uniquement par le super-utilisateur

Contrairement à ce que leur nom évoque, c'est le fichier /etc/shadow qui contient les mots de passe et non pas le fichier /etc/passwd.

Groupes d'utilisateurs

Principe :

- Rassembler les utilisateurs dans des **groupes**
- Donner des **permissions** à chaque groupe

Exemple : “tous les membres de ce projet ont le droit d'écrire dans ce répertoire”

- On met tous les membres du projet dans un groupe
- On donne les droits en écriture sur le répertoire à ce groupe

Chaque utilisateur fait partie d' **au moins 1 groupe** :

- Obligatoirement dans un **groupe primaire** et un seul
- Éventuellement dans des **groupes supplémentaires**

Commande id :

```
coti@maximum:~$ id figatellix
uid=1000(figatellix) gid=1001(figatellix) groupes=24(cdrom),25(floppy),
29(audio),30(dip),44(video),46(plugdev),104(scanner),109(bluetooth),
111(netdev),1001(figatellix)
```

Informations sur les groupes

Fichier `/etc/group` : contient les définitions des groupes

- nom du groupe ;
- x, anciennement le mot de passe, non-affichable
- le numéro du groupe
- la liste des utilisateurs faisant partie du groupe en tant que groupe secondaire

```
root:x:0:
cdrom:x:24:figatellix
audio:x:29:pulse,figatellix
figatellix:x:1001:
```

Informations sur les groupes

Fichier `/etc/group` : contient les définitions des groupes

- nom du groupe ;
- x, anciennement le mot de passe, non-affichable
- le numéro du groupe
- la liste des utilisateurs faisant partie du groupe en tant que groupe secondaire

```
root:x:0:
cdrom:x:24:figatellix
audio:x:29:pulse,figatellix
figatellix:x:1001:
```

Fichier `/etc/gshadow` : contient les mots de passe des groupes.

- nom du groupe
- mot de passe du groupe, crypté (cf fichier `/etc/shadow`)
- liste des utilisateurs définis comme administrateurs de ce groupe
- liste des utilisateurs faisant partie du groupe en tant que groupe secondaire

```
root:*::
cdrom:*::figatellix
audio:*::pulse,figatellix
figatellix:!*::
```

Modifier les groupes

Plusieurs possibilités pour manipuler quels utilisateurs sont dans les groupes :

- Commande `usermod` : modification du groupe d'un utilisateur

```
root@maximum:~# usermod -a -G backup figatellix
```
- Pour modifier les groupes secondaires : modification du fichier `/etc/group`, ajout/suppression des utilisateurs dans les groupes
- Pour modifier le groupe primaire : modification du fichier `/etc/passwd`

Modification des propriétés d'un groupe : commande `groupmod`

- Modification son gid, son mot de passe ou son nom du groupe.

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs**
 - Permissions Unix
 - Devenir super-utilisateur
 - Quotas d'espace de stockage
 - Limites du système d'exploitation
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs**
 - Permissions Unix
 - Devenir super-utilisateur
 - Quotas d'espace de stockage
 - Limites du système d'exploitation
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées

Permissions associées aux fichiers

```
coti@maximum:~/repertoire$ ls -l
total 16
lrwxrwxrwx 1 coti users  27 avril 27 16:48 lien -> /boot/vmlinuz-3.2.0
-rwxr-xr-x 1 coti users 5837 avril 27 16:48 plante
-rw-r--r-- 1 coti users  225 avril 27 16:48 plante.c
drwxr-xr-x 2 coti users 4096 avril 27 16:48 subdir
```

1ere colonne : **mode** de chaque fichier

- Composé de dix caractères
- un **indicateur de type** (un caractère) : répertoire, lien symbolique, pipeline, socket, fichier normal...
- puis les **droits d'accès** à ce fichier (neuf caractères).

Permissions associées aux fichiers

```
coti@maximum:~/repertoire$ ls -l
total 16
lrwxrwxrwx 1 coti users 27 avril 27 16:48 lien -> /boot/vmlinuz-3.2.0
-rwxr-xr-x 1 coti users 5837 avril 27 16:48 plante
-rw-r--r-- 1 coti users 225 avril 27 16:48 plante.c
drwxr-xr-x 2 coti users 4096 avril 27 16:48 subdir
```

1ere colonne : **mode** de chaque fichier

- Composé de dix caractères
- un **indicateur de type** (un caractère) : répertoire, lien symbolique, pipeline, socket, fichier normal...
- puis les **droits d'accès** à ce fichier (neuf caractères).

Droits d'accès : 3 blocs de 3 caractères :

- Droits de l'utilisateur propriétaire du fichier
- Droits du groupe propriétaire du fichier
- Droits du reste du monde

Permissions associées aux fichiers

Permissions : 3 caractères par catégorie concernée :

- Lecture : r (read)
- Écriture : w (write)
- Exécution : x (execute)

Exemple : `rwxr-xr-x`

- Le **propriétaire** du fichier a le droit d' **exécuter, écrire et lire** le fichier (droits `rwX`)
- Les **membres du groupe** propriétaire du fichier ont le droit de l' **exécuter et de le lire** mais pas d'écrire dessus (droits `r-x`)
- Le **reste du monde** a le droit de l' **exécuter et de le lire** mais pas d'écrire dessus (droits `r-x`)

Permissions associées aux fichiers

Autre façon d'exprimer les permissions associées à un fichier : **forme numérique**

- Chaque caractère = 1 bit
 - Valeur 1 si la permission est présente, 0 sinon
- 3 groupes de 3 bits = 3 nombres codés sur 3 bits chacun

Exemple : `rwrx-x-x`

- En binaire : 111 101 001
- En décimal : 751

Modification des permissions

Modification des permissions associées à un fichier : **commande chmod**
(*change mode*)

Utilisation :

- Définition des droits : signe =

```
coti@maximum:~/repertoire$ chmod =rwxrwxrwx plante.c
```

- Ajout ou retrait de droits : signe + ou -

```
coti@maximum:~/repertoire$ chmod g-x plante.c
```

```
coti@maximum:~/repertoire$ chmod g+w plante.c
```

- Utilisation de la forme numérique

```
coti@maximum:~/repertoire$ chmod 751 plante.c
```

Masque de création

Masque de création = définit permissions associées à un fichier à sa création

Quatre chiffres :

- Indicateur de type de fichier (donc 0)
- Masque de permissions

On calcule les permissions en retranchant ce masque :

- De 0777 pour un répertoire
- De 0666 pour un fichier normal

Exemple :

```
coti@maximum:~/repertoire$ umask  
0022
```

- Création d'un répertoire : $0777 - 0022 = 0755 = \text{rwxr-xr-x}$
- Création d'un fichier normal : $0666 - 0022 = 0644 = \text{rw-r-r-}$

Propriétaire d'un fichier

Lorsqu'un fichier est créé, quels sont son utilisateur et son groupe propriétaires ?

- Utilisateur propriétaire : **utilisateur qui a créé le fichier**
- Groupe propriétaire : **groupe primaire** de l'utilisateur qui a créé le fichier

```
coti@maximum:/tmp$ touch toto
coti@maximum:/tmp$ ls -l toto
-rw-r--r-- 1 coti 1000 331 avril 27 16:39 toto
```

Changement de propriétaire d'un fichier : **commande chown** (*change owner*)

- Changement d'utilisateur propriétaire ou également du groupe propriétaire

```
root@maximum:/tmp# chown figatellix toto
root@maximum:/tmp# ls -l toto
-rw-r--r-- 1 figatellix 1000 331 avril 27 16:39 toto
root@maximum:/tmp# chown figatellix:audio toto
root@maximum:/tmp# ls -l toto
-rw-r--r-- 1 figatellix audio 331 avril 27 16:39 toto
```

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs**
 - Permissions Unix
 - **Devenir super-utilisateur**
 - Quotas d'espace de stockage
 - Limites du système d'exploitation
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées

Devenir super-utilisateur

Deux possibilités :

- **Changement d'identité** : commande `su` (*switch user*)
 - Permanent : on *devient* le super-utilisateur
 - Nécessite de connaître le mot de passe
- Exécution d'une commande **en tant que super-utilisateur** : commande `sudo`
 - Temporaire : uniquement le temps d'exécuter la commande
 - Nécessite d'avoir été autorisé explicitement à utiliser `sudo` (fichier `/etc/sudoers`)
 - Actions journalisées : on sait qui a fait quoi en tant que super-utilisateur
 - Possibilité de définir précisément ce qu'un utilisateur a le droit de faire avec `sudo`

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs**
 - Permissions Unix
 - Devenir super-utilisateur
 - **Quotas d'espace de stockage**
 - Limites du système d'exploitation
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées

Principe des quotas

Quotas : mis en place au niveau du **système de fichiers**

- Limitation de l'espace de stockage que peut occuper un utilisateur
- Deux éléments limités : l' **espace occupé** et le **nombre de fichiers**

Deux limites :

- La *limite dure* (hard limit), qui ne peut pas être dépassée
- La *limite douce* (soft limit), qui peut être dépassée temporairement

Activation des quotas sur une partition

- 1 La partition doit être **montée avec les options idoines** :

```
/dev/sdb1 /home ext3 defaults,usrquota,grpquota 1 1
```

- 2 Deux **fichiers à la racine** de la partition concernée :

- quota.user
- quota.group

Il faut juste qu'ils soient présents et aient les droits 600.

- 3 Activation et désactivation des quotas :

```
coti@maximum:~$ sudo quotaon -a
```

```
coti@maximum:~$ sudo quotaoff
```

Vérification des quotas :

```
coti@maximum:~$ sudo quotacheck -vguma -F vfstv0
```

Configuration des quotas

Commande edquota : **edite les quotas** sur une partition donnée.

Ouvre un éditeur de texte qui affiche un tableau modifiable :

```
coti@maximum:~$ sudo edquota -u user1
```

```
Quotas disque pour user user1 (uid 1005) :
```

Système de fichiers	blocs	souple	stricte	inodes
/dev/sdb5	24	150000	0	13

Configuration des quotas

Commande edquota : **edite les quotas** sur une partition donnée.

Ouvre un éditeur de texte qui affiche un tableau modifiable :

```
coti@maximum:~$ sudo edquota -u user1
Quotas disque pour user user1 (uid 1005) :
Système de fichiers      blocs      souple      stricte      inodes
/dev/sdb5                24         150000     0            13
```

Affichage des quotas et **état de l'utilisation** : commande repquota

```
coti@maximum:~$ sudo repquota /home
*** Rapport pour les quotas user sur le périphérique /dev/sdb5
Période de sursis bloc : 7days ; période de sursis inode : 7days
                Block limits                File limits
Utilisateur    utilisé souple stricte sursis utilisé souple stricte su
-----
root           --      20      0      0              2      0      0
user1          --      24     150      0             13      0      0
user2          --      16      0      0              4      0      0
```

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs**
 - Permissions Unix
 - Devenir super-utilisateur
 - Quotas d'espace de stockage
 - **Limites du système d'exploitation**
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées

Limites du système d'exploitation

Fixées par le **système d'exploitation**

- Garde-fou pour l'utilisateur (en cas de bug par exemple)
- Pour l'administrateur : évite qu'un utilisateur ne monopolise le système

Deux sortes de limites :

- La **limite douce** (soft limit), fixée par l'utilisateur ;
- La **limite dure** (hard limit), fixée par l'administrateur.

Commande ulimit

Limites du système d'exploitation

Affichage des limites : `ulimit -a (all)`

```
core file size          (blocks, -c) 0
data seg size          (kbytes, -d) unlimited
scheduling priority    (-e) 0
file size              (blocks, -f) unlimited
pending signals        (-i) 63602
max locked memory      (kbytes, -l) 64
max memory size        (kbytes, -m) unlimited
open files             (-n) 1024
pipe size              (512 bytes, -p) 8
POSIX message queues   (bytes, -q) 819200
real-time priority     (-r) 0
stack size            (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
max user processes    (-u) 63602
virtual memory         (kbytes, -v) unlimited
file locks            (-x) unlimited
```

Limites du système d'exploitation

Configuration des limites système : fichier `/etc/security/limits.conf`, par groupe ou par utilisateur

```
@etudiants    hard    nprocs    1024
@etudiants    soft    nprocs    512
```

Configuration des limites utilisateur : commande `ulimit` avec l'option correspondant à la limite à modifier

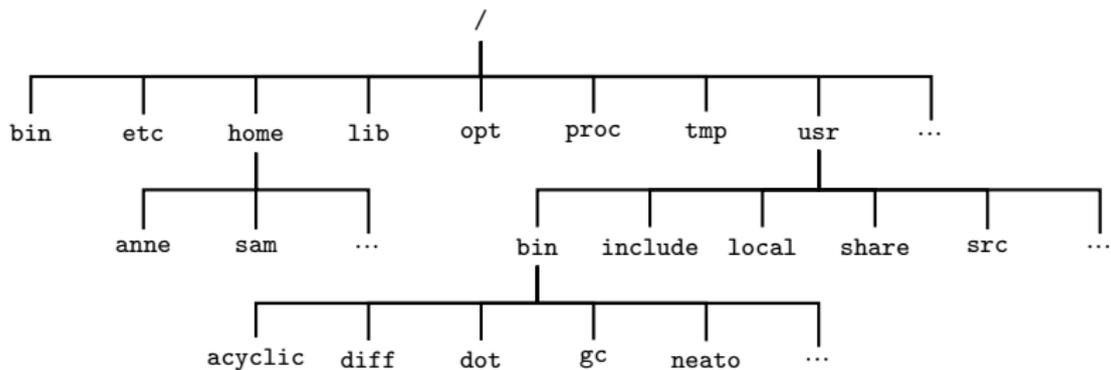
```
coti@maximum:~$ ulimit -n
1024
coti@maximum:~$ ulimit -n 2086
coti@maximum:~$ ulimit -n
2086
```

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers**
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

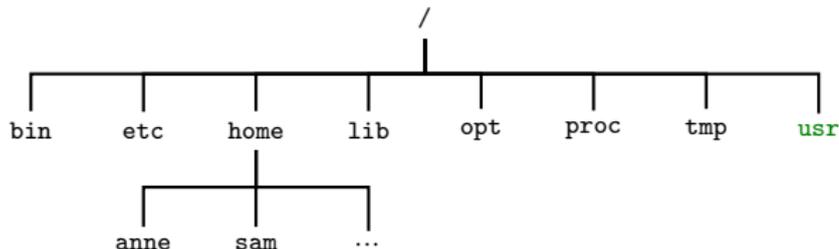
Arborescence de fichiers

Dans les systèmes Unix : structure arborescente unique



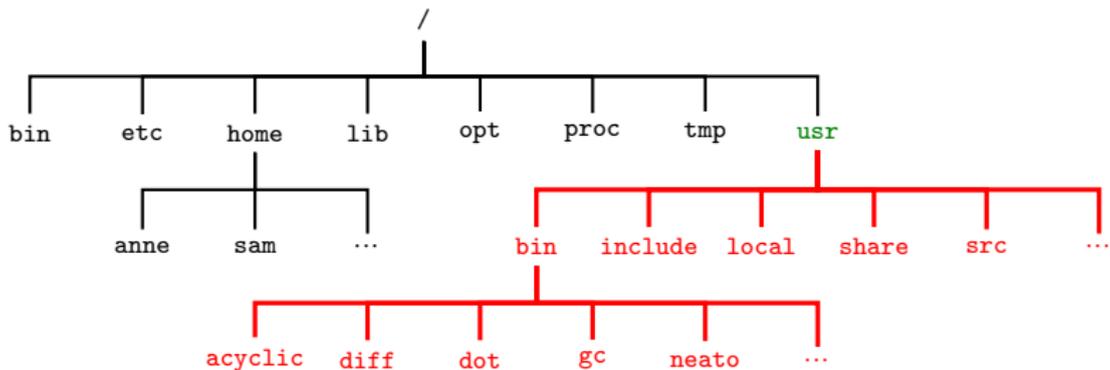
Rattachement d'une partition

Pour rattacher une partition à cette structure, on la **monte** à un **point de montage** :



Rattachement d'une partition

Pour rattacher une partition à cette structure, on la **monte** à un **point de montage** :



Définition des partitions à monter au démarrage

Fichier `/etc/fstab`

- Donne les partitions, leur type et leur point de montage

```
# <file system> <mount point> <type> <options> <dump> <pass>
/dev/sda1 / ext3 errors=remount-ro 0 1
/dev/sdb5 /home ext3 defaults 0 2
/dev/sdb1 /opt ext3 defaults 0 2
/dev/sda6 /usr ext3 defaults 0 2
/dev/sda7 /var ext3 defaults 0 2
/dev/sda5 none swap sw 0 0
/dev/sr0 /media/cdrom0 udf,iso9660 user,noauto 0 0
```

Montage à l'exécution

Pour voir **quelles partitions sont montées** : commande mount

```
coti@thorim:~$ mount
/dev/disk0s2 on / (hfs, local, journaled)
devfs on /dev (devfs, local, nobrowse)
map -hosts on /net (autofs, nosuid, automounted, nobrowse)
map auto_home on /home (autofs, automounted, nobrowse)
```

Montage à l'exécution

Pour voir **quelles partitions sont montées** : commande `mount`

```
coti@thorim:~$ mount
/dev/disk0s2 on / (hfs, local, journaled)
devfs on /dev (devfs, local, nobrowse)
map -hosts on /net (autofs, nosuid, automounted, nobrowse)
map auto_home on /home (autofs, automounted, nobrowse)
```

Montage d'une partition à la main : commande `mount` (avec des arguments) :

```
mount [options] <partition> <point de montage>
```

Par exemple :

```
root@maximum:~# mount /dev/sdb5 /home
```

Montage à l'exécution

Pour voir **quelles partitions sont montées** : commande `mount`

```
coti@thorim:~$ mount
/dev/disk0s2 on / (hfs, local, journaled)
devfs on /dev (devfs, local, nobrowse)
map -hosts on /net (autofs, nosuid, automounted, nobrowse)
map auto_home on /home (autofs, automounted, nobrowse)
```

Montage d'une partition à la main : commande `mount` (avec des arguments) :

```
mount [options] <partition> <point de montage>
```

Par exemple :

```
root@maximum:~# mount /dev/sdb5 /home
```

Démontage d'une partition : commande `umount`

```
root@maximum:~# umount /home
```

Nommage des disques sous Linux

Rappel : sous Unix, tout est fichier

- On peut voir les partitions comme des fichiers spéciaux, dans l'arborescence

On trouve ces fichiers dans le répertoire `/dev`

Systeme de nommage des disques :

- **Première lettre** : type de connectique
 - `h` pour une interface IDE, `s` pour une interface SATA ou SCSI.
- **Deuxième lettre** : un `d`.
- **Troisième lettre** sert à numéroter les disques : le premier utilise un `a`, le deuxième un `b`, etc...

Exemples :

- `/dev/hda` = premier disque IDE
- `/dev/sdc` = troisième disque SATA ou SCSI

Nommage des partitions sous Linux

Un disque doit être **partitionné** pour être utilisé.

Nommage sous Linux :

- Nom du disque dont la partition fait partie
- Numéro de partition, en commençant à 1

Exemple : `/dev/sda3` = troisième partition du premier disque SATA ou SCSI.

Système de fichiers : organise le **stockage et l'accès des données** sur une partition d'un disque.

Fonctionnalités de base d'un système de fichiers :

- Ouverture et fermeture de fichiers
- Gestion des droits d'accès...

Exemples :

- FAT (puis FAT16, FAT32, VFAT...) : système historique sous MS DOS puis Windows
- NTFS : remplace FAT, permet de gérer les droits d'accès
- ext2, ext3, ext4 : utilisés par défaut par les systèmes Linux
- HFS, HFS+ : utilisé sous Mac OS X, iPod, iPad, iPhone...

Mise en place d'un système de fichiers

La commande `mkfs` permet de **mettre en place un système de fichiers** sur une partition

- On spécifie le type de système de fichiers

Exemple :

```
root@maximum:~# mkfs -t ext3 /dev/sda5
```

Ici : on crée un système de fichiers `ext3` sur la partition `/dev/sda5`.

Mise en place d'un système de fichiers

La commande `mkfs` permet de **mettre en place un système de fichiers** sur une partition

- On spécifie le type de système de fichiers

Exemple :

```
root@maximum:~# mkfs -t ext3 /dev/sda5
```

Ici : on crée un système de fichiers `ext3` sur la partition `/dev/sda5`.

Redimensionner un système de fichiers : `resize2fs`

- Attention en diminuant la taille : les données présentes doivent continuer à tenir !
- La partition ne doit pas être montée dans l'arborescence de fichiers au moment de l'exécution

Mise en place d'un système de fichiers

La commande `mkfs` permet de **mettre en place un système de fichiers** sur une partition

- On spécifie le type de système de fichiers

Exemple :

```
root@maximum:~# mkfs -t ext3 /dev/sda5
```

Ici : on crée un système de fichiers `ext3` sur la partition `/dev/sda5`.

Redimensionner un système de fichiers : `resize2fs`

- Attention en diminuant la taille : les données présentes doivent continuer à tenir !
- La partition ne doit pas être montée dans l'arborescence de fichiers au moment de l'exécution

Vérification de l'intégrité d'un système de fichiers, **réparation d'un système journalisé** : `e2fsck`

- La partition ne doit pas être montée dans l'arborescence de fichiers au moment de l'exécution

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr**
 - Stockage des données sur un ordinateur
 - Fiabilité
 - Techniques de stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr**
 - Stockage des données sur un ordinateur
 - Fiabilité
 - Techniques de stockage sûr
- 7 Métrologie
- 8 Tâches planifiées

Systèmes de stockage

Types de mémoire trouvés sur une machine :

- De la mémoire **volatile** , utilisée par les applications en cours d'exécution
- De la mémoire **non volatile** , utilisée pour stocker des données de façon permanente

Systèmes de stockage

Types de mémoire trouvés sur une machine :

- De la mémoire **volatile** , utilisée par les applications en cours d'exécution
- De la mémoire **non volatile** , utilisée pour stocker des données de façon permanente

La mémoire non volatile est-elle vraiment sûre ?

- Exempte de pannes matérielles ? → **NON !**

Problème : comment **conserver de façon sûre** les données de la mémoire non volatile ?

Types de mémoire sur une machine

Mémoires volatiles :

- Flux d'instructions utilisées par le programme en cours : **cache L1** (ou cache de niveau 1)
 - Très rapide, situé sur le processeur, le plus proche de l'ALU, très petit (env. 32ko/cœur)

Types de mémoire sur une machine

Mémoires volatiles :

- Flux d'instructions utilisées par le programme en cours : **cache L1** (ou cache de niveau 1)
 - Très rapide, situé sur le processeur, le plus proche de l'ALU, très petit (env. 32ko/cœur)
- Pour ce qui ne tient pas en cache L1 : **cache L2**
 - Rapide, situé proche du cœur mais plus loin de l'ALU, plus gros que le L1 (env. 256ko/cœur)

Types de mémoire sur une machine

Mémoires volatiles :

- Flux d'instructions utilisées par le programme en cours : **cache L1** (ou cache de niveau 1)
 - Très rapide, situé sur le processeur, le plus proche de l'ALU, très petit (env. 32ko/cœur)
- Pour ce qui ne tient pas en cache L1 : **cache L2**
 - Rapide, situé proche du cœur mais plus loin de l'ALU, plus gros que le L1 (env. 256ko/cœur)
- Pour ce qui ne tient pas en cache L2 : **cache L3**
 - Plus lent que le cache L2, situé sur le processeur, généralement un seul par processeur, plus gros que L2 (env. 8Mo/proc)

Types de mémoire sur une machine

Mémoires volatiles :

- Flux d'instructions utilisées par le programme en cours : **cache L1** (ou cache de niveau 1)
 - Très rapide, situé sur le processeur, le plus proche de l'ALU, très petit (env. 32ko/cœur)
- Pour ce qui ne tient pas en cache L1 : **cache L2**
 - Rapide, situé proche du cœur mais plus loin de l'ALU, plus gros que le L1 (env. 256ko/cœur)
- Pour ce qui ne tient pas en cache L2 : **cache L3**
 - Plus lent que le cache L2, situé sur le processeur, généralement un seul par processeur, plus gros que L2 (env. 8Mo/proc)
- Pour tout ce qui ne tient pas en cache : **mémoire vive**
 - Plus lent que les caches, située sur la carte mère à l'extérieur du processeur, gros (env. 4 à 512 Go/machine)

Types de mémoire sur une machine

Mémoires volatiles :

- Flux d'instructions utilisées par le programme en cours : **cache L1** (ou cache de niveau 1)
 - Très rapide, situé sur le processeur, le plus proche de l'ALU, très petit (env. 32ko/cœur)
- Pour ce qui ne tient pas en cache L1 : **cache L2**
 - Rapide, situé proche du cœur mais plus loin de l'ALU, plus gros que le L1 (env. 256ko/cœur)
- Pour ce qui ne tient pas en cache L2 : **cache L3**
 - Plus lent que le cache L2, situé sur le processeur, généralement un seul par processeur, plus gros que L2 (env. 8Mo/proc)
- Pour tout ce qui ne tient pas en cache : **mémoire vive**
 - Plus lent que les caches, située sur la carte mère à l'extérieur du processeur, gros (env. 4 à 512 Go/machine)
- Pour tout ce qui ne tient pas en mémoire vive : **swap**
 - Partition sur le disque dur, taille 1 ou 2 fois la taille de la mémoire vive, très lent !

Types de mémoire sur une machine

Mémoires volatiles :

- Flux d'instructions utilisées par le programme en cours : **cache L1** (ou cache de niveau 1)
 - Très rapide, situé sur le processeur, le plus proche de l'ALU, très petit (env. 32ko/cœur)
- Pour ce qui ne tient pas en cache L1 : **cache L2**
 - Rapide, situé proche du cœur mais plus loin de l'ALU, plus gros que le L1 (env. 256ko/cœur)
- Pour ce qui ne tient pas en cache L2 : **cache L3**
 - Plus lent que le cache L2, situé sur le processeur, généralement un seul par processeur, plus gros que L2 (env. 8Mo/proc)
- Pour tout ce qui ne tient pas en cache : **mémoire vive**
 - Plus lent que les caches, située sur la carte mère à l'extérieur du processeur, gros (env. 4 à 512 Go/machine)
- Pour tout ce qui ne tient pas en mémoire vive : **swap**
 - Partition sur le disque dur, taille 1 ou 2 fois la taille de la mémoire vive, très lent !

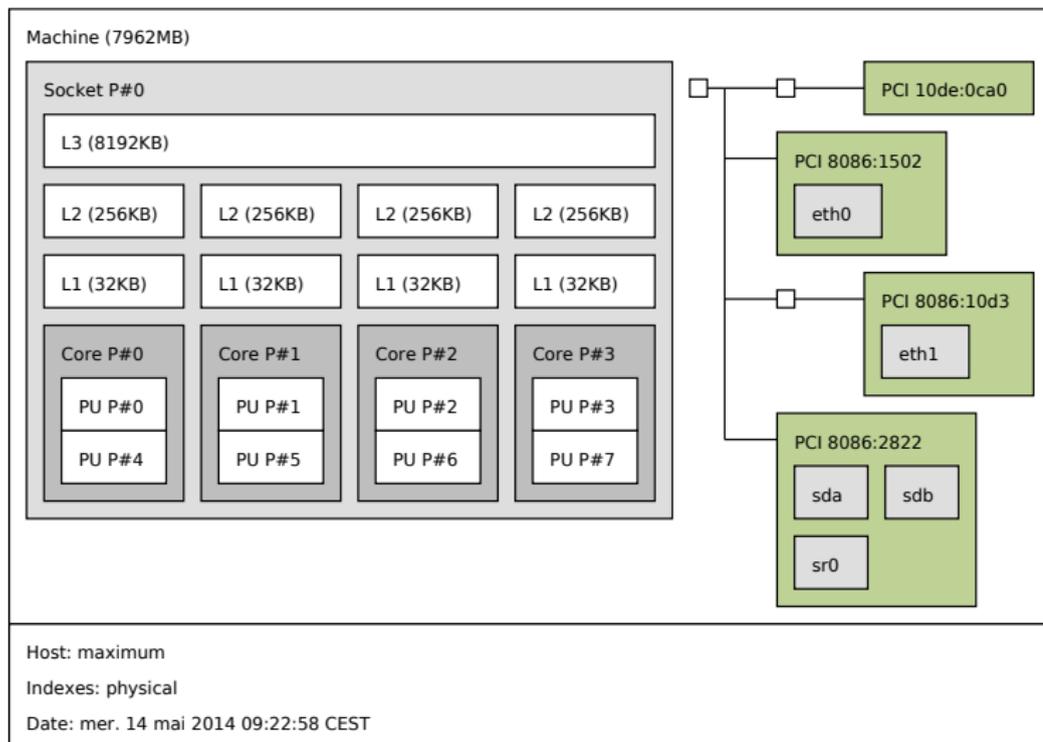
Rapidité :

- L1 > L2 > L3 > RAM > swap

Taille :

- L1 < L2 < L3 < RAM < swap

Hwloc



Non-volatilité

Les données en mémoire volatile **ne sont pas conservées** quand la machine est éteinte

→ Stockage des données en **mémoire non-volatile**

Non-volatilité

Les données en mémoire volatile **ne sont pas conservées** quand la machine est éteinte

→ Stockage des données en **mémoire non-volatile**

Types de **mémoire non volatile** :

- Disque dur mécanique, SSD, bandes magnétiques...

Comment sont stockées les données physiquement ?

- **Disque dur mécanique** : des plateaux en rotation autour d'un axe, des têtes de lecture se déplaçant
 - Grosse partie **mécanique** !
 - Fragile et lent
- **SSD** (Solid State device) : mémoire flash
 - Aucune partie mécanique, donc robuste, rapide et économe en énergie
 - Petite taille
 - Les cellules ont une durée de vie limitée

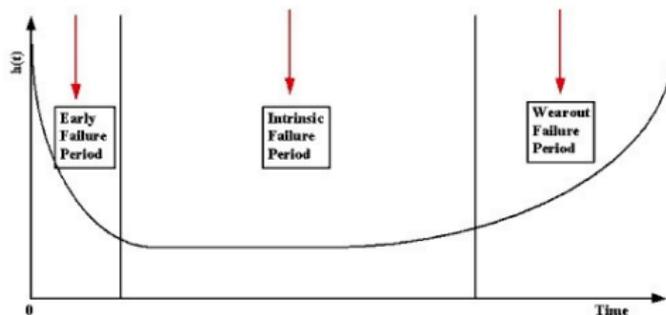
Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr**
 - Stockage des données sur un ordinateur
 - **Fiabilité**
 - Techniques de stockage sûr
- 7 Métrologie
- 8 Tâches planifiées

Fiabilité

Les disques durs mécaniques sont **peu fiables**

- Pannes au début pendant les 3 premiers mois (période de rodage), pannes au bout de plusieurs années (3-5 ans : usure)
- Probabilité de pannes : courbe en baignoire



Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr**
 - Stockage des données sur un ordinateur
 - Fiabilité
 - Techniques de stockage sûr
- 7 Métrologie
- 8 Tâches planifiées

Techniques de stockage sûr

Idée de base : **pouvoir récupérer les données après une panne matérielle**
(disque dur HS...)

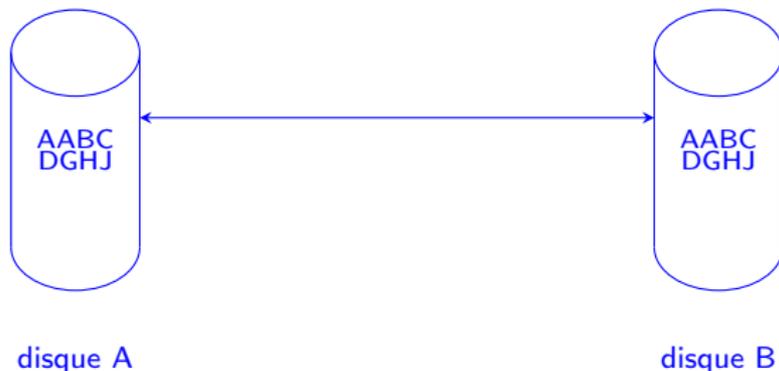
Deux catégories :

- Réplication : on recopie **l'intégralité** des données
- Redondance : on ajoute de la redondance pour **reconstruire** les données perdues

Réplication

Réplication : **copie pure et simple des données**

- On a des données sur un disque, on les recopie intégralement sur l'autre disque



- **En cas de panne** : on prend l'autre disque, on a directement l'intégralité des données
- Inconvénient : pour supporter N pannes, on a besoin de **multiplier l'espace de stockage** par $N+1$

Réplication : sous Unix

dd : copie brute d'un fichier dans un autre fichier

- Peut être utilisé entre deux partitions, entre deux disques, d'une partition dans un fichier...

```
root@maximum:~# dd if=/dev/sda of=/dev/sdb bs=1024
```

```
root@maximum:~# dd if=/dev/sda of=svg_date.img bs=1024
```

```
root@maximum:~# dd if=/dev/sda bs=1024 | gzip > svg_date.img.gz
```

tar : rassembler des fichiers dans une archive, en conservant les propriétés associées aux fichiers

- options `-posix -numeric-owner`

```
coti@maximum:~$ tar -posix --numeric-owner czf /opt > /tempo/svg.tgz
```

Sauvegarde incrémentale

Idée : ne sauvegarder **que les modifications effectuées depuis la dernière fois**

- Diminution de la taille de la sauvegarde
- Reconstruction des données étape par étape
- Nécessite une sauvegarde complète de temps en temps

Notion de **niveau de sauvegarde** : par rapport à quelle sauvegarde incrémentale considère-t-on les modifications ?

- Depuis la dernière sauvegarde : niveau 1
- Depuis l'avant-dernière sauvegarde : niveau 2
- Depuis N sauvegardes : sauvegarde de niveau N
- Sauvegarde intégrale : niveau 0

Sauvegarde incrémentale : exemple

On a les données :

AABBCCDDEEFFGGHHIIJJKK

On sauvegarde l'intégralité :

AABBCCDDEEFFGGHHIIJJKK

Sauvegarde incrémentale : exemple

On a les données :

```
AABBCCDDEEFFGGHHIIJJKK
```

On sauvegarde l'intégralité :

```
AABBCCDDEEFFGGHHIIJJKK
```

On modifie les données : on remplace FF par OO à 10 caractères du début

```
AABBCCDDEEOOGGHHIIJJKK
```

On sauvegarde uniquement les modifications. Notons par exemple

```
+10 : -FF+OO
```

Sauvegarde incrémentale : exemple

On a les données :

```
AABBCCDDEEFFGGHHIIJJKK
```

On sauvegarde l'intégralité :

```
AABBCCDDEEFFGGHHIIJJKK
```

On modifie les données : on remplace FF par OO à 10 caractères du début

```
AABBCCDDEEOGGHHIIJJKK
```

On sauvegarde uniquement les modifications. Notons par exemple

```
+10 : -FF+OO
```

Nouvelle modification :

```
AAIICCDDEEOGGHUUIJJKK
```

Sauvegarde de niveau 1 :

```
+2 : -BB+II
```

```
+15 : -HI+UU
```

Sauvegarde incrémentale : exemple

Panne ! On reconstitue les données en prenant **la dernière sauvegarde intégrale** et en lui appliquant **les sauvegardes incrémentales** effectuées depuis :

AABBCCDDEEFFGGHHIIJJKK

+10 : -FF+00

-> AABBCCDDEEOGGHHIIJJKK

+2 : -BB+II

+15 : -HI+UU

-> AAIIICCDDEEOGGHUUIJJKK

On obtient bien les données dans leur état au moment de la dernière sauvegarde incrémentale.

Mise en place sous Unix avec dump

`dump` : permet de faire des sauvegardes incrémentales (répertoire, système de fichiers...) en spécifiant le niveau de sauvegarde

```
root@abidjan:~# dump -0uf data0.bkp /home
root@abidjan:~# dump -1uf data1.bkp /home
root@abidjan:~# dump -2uf data2.bkp /home
root@abidjan:~# dump -3uf data3.bkp /home
```

Taille des sauvegardes obtenues :

```
root@abidjan:~# ls -l
total 18322096
-rw-r--r-- 1 root root      2764800 avril 26 00:46 data1.bkp
-rw-r--r-- 1 root root      2764800 avril 26 00:50 data2.bkp
-rw-r--r-- 1 root root 18737960960 avril 26 00:45 data.bkp
```

`restore` permet de restaurer l'état à partir des sauvegardes générées par `dump`

- On applique les sauvegardes incrémentales l'une après l'autre

Redondance : le RAID

RAID : Redundant Array of Independent Disks, ou Redundant Array of Inexpensive Disks

- But : obtenir un système de stockage sûr et rapide à partir de disques bon marché et donc peu fiables et peu performants

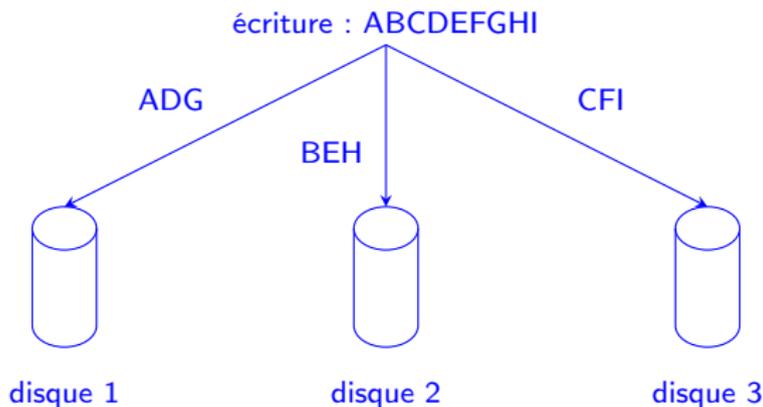
Principes du RAID :

- Possibilité de **lire sur plusieurs disques à la fois**
- Possibilité **d'écrire sur plusieurs disques à la fois**
- Possibilité **d'écrire autant de fois la même information** , ou de **découper l'information et la lire ou l'écrire en parallèle sur plusieurs disques**
- Possibilité **d'écrire une information plusieurs fois** , ou de **récupérer une information en combinant plusieurs**

RAID 0

RAID 0 : lecture et écriture en parallèle sur plusieurs disques

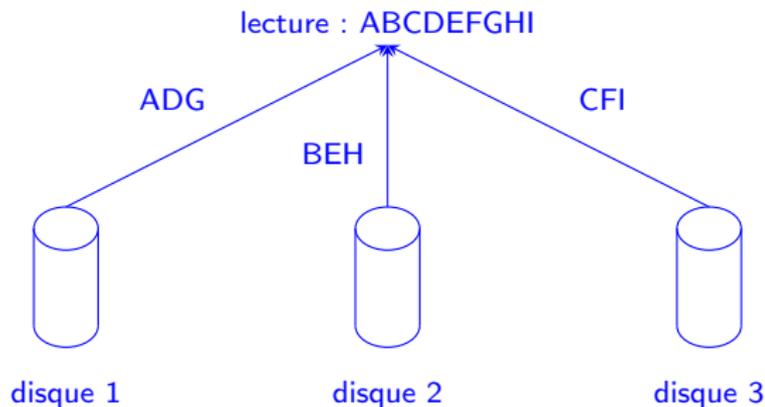
- Les données sont présentes une seule fois dans le système, mais elles sont découpées entre plusieurs disques
- Le stockage n'est **pas fiabilisé**
- Plus rapide !



RAID 0

On écrit un volume V sur D disques :

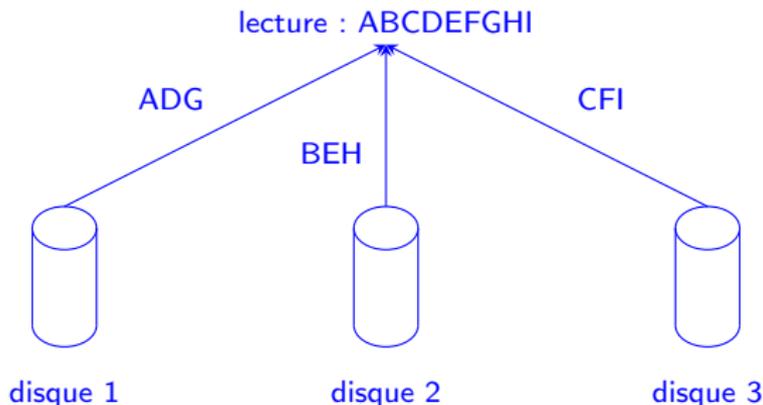
- chaque disque recevra le volume V/D
- si on écrit à la vitesse de τ octets par seconde : idéalement, le temps de lecture sera de $V/(D * \tau)$ au lieu de V/τ



RAID 1

RAID 1 : réplication pure et simple des données

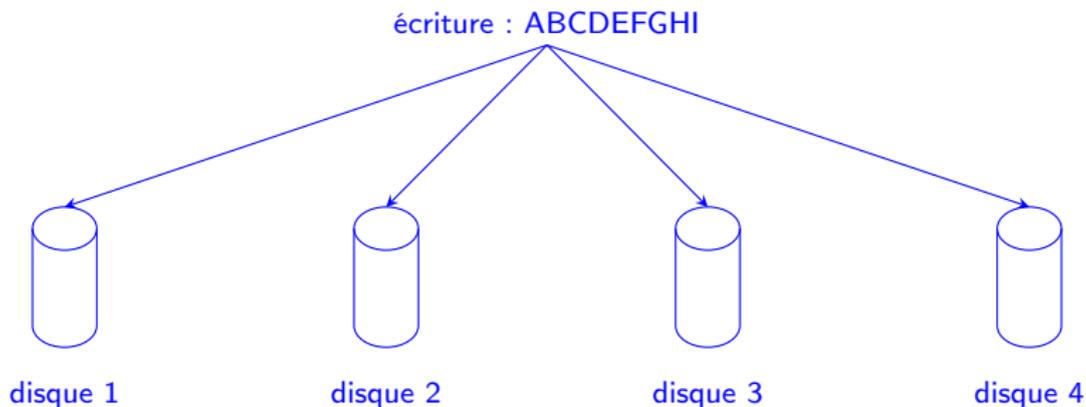
- On écrit la même chose sur plusieurs disques en parallèle
- Stockage fiabilisé par redondance
- Pas plus rapide en écriture
- En lecture, deux possibilités :
 - le disque le moins chargé répond le plus rapidement : équilibrage de charge, un peu plus rapide
 - tirage au sort : on lit aléatoirement sur l'un ou sur l'autre. Répartition de charge



RAID 5

RAID 5 : combine la réplication et la lecture/écriture parallèle

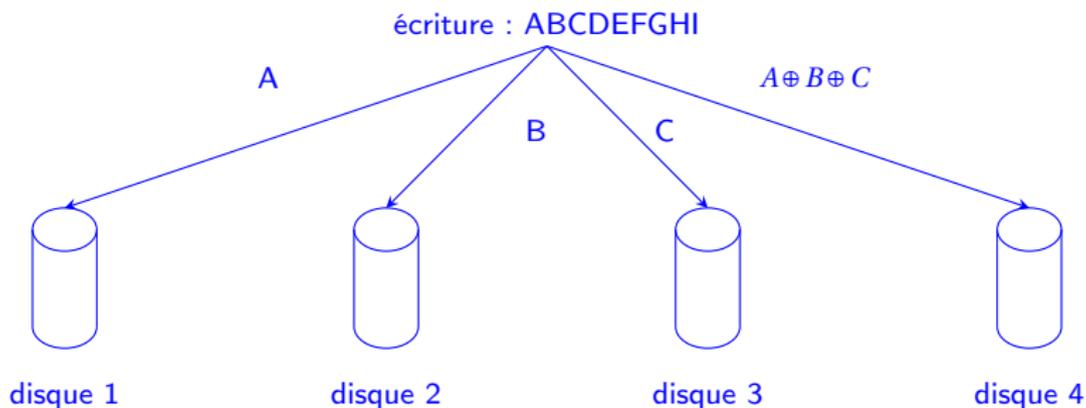
- Basé sur un système de détection et correction d'erreur par parité
- On réserve 1 ou plusieurs disque(s) pour recevoir un bloc de parité
- Lecture/écriture en parallèle sur les disques
- Si un disque tombe en panne, on reconstitue les données perdues avec les données restantes et un bloc de parité



RAID 5

RAID 5 : combine la réplication et la lecture/écriture parallèle

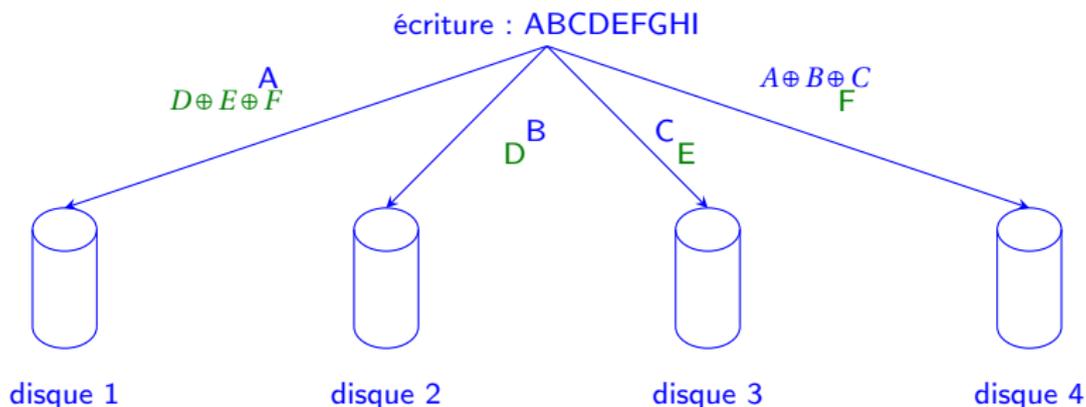
- Basé sur un système de détection et correction d'erreur par parité
- On réserve 1 ou plusieurs disque(s) pour recevoir un bloc de parité
- Lecture/écriture en parallèle sur les disques
- Si un disque tombe en panne, on reconstitue les données perdues avec les données restantes et un bloc de parité



RAID 5

RAID 5 : combine la réplication et la lecture/écriture parallèle

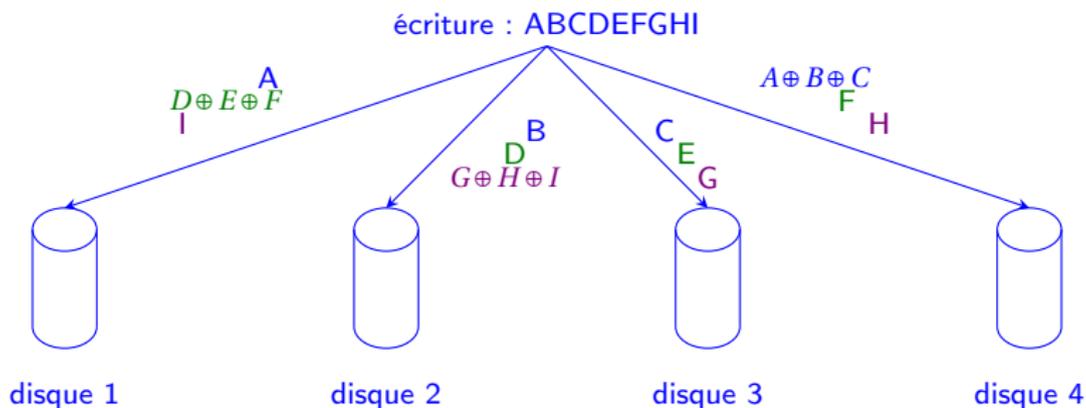
- Basé sur un système de détection et correction d'erreur par parité
- On réserve 1 ou plusieurs disque(s) pour recevoir un bloc de parité
- Lecture/écriture en parallèle sur les disques
- Si un disque tombe en panne, on reconstitue les données perdues avec les données restantes et un bloc de parité



RAID 5

RAID 5 : combine la réplication et la lecture/écriture parallèle

- Basé sur un système de détection et correction d'erreur par parité
- On réserve 1 ou plusieurs disque(s) pour recevoir un bloc de parité
- Lecture/écriture en parallèle sur les disques
- Si un disque tombe en panne, on reconstitue les données perdues avec les données restantes et un bloc de parité



- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie**
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

Sondes matérielles et logicielles

- Fournies par le système d'exploitation : donnent une idée sur ce qui est en train de tourner
- Surveillance de l'état des ressources
- Pseudo-système de fichiers /proc
- Utilisation de capteurs de températures

Charge CPU : commande top

- Donne la liste des processus exécutés sur la machine
- Pour chaque processus, quelques infos : pourcentage CPU utilisé, mémoire utilisée, temps écoulé depuis le début de l'exécution...
- Possibilité de trier selon un critère en particulier

```
top - 14:47:39 up 14 days, 2:35, 10 users, load average: 0,96, 0,99, 0,92
Tasks: 244 total, 2 running, 242 sleeping, 0 stopped, 0 zombie
%Cpu(s): 2,3 us, 0,5 sy, 0,0 ni, 96,8 id, 0,5 wa, 0,0 hi, 0,0 si, 0,0 st
KiB Mem: 8153328 total, 7021852 used, 1131476 free, 196668 buffers
KiB Swap: 7811068 total, 2644 used, 7808424 free, 2781416 cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
28710	coti	20	0	986m	148m	23m	S	7,0	1,9	0:36.82	chrome
3529	coti	20	0	1189m	68m	11m	S	3,7	0,9	35:04.63	chrome
2989	coti	9	-11	434m	8824	5584	S	2,3	0,1	22:57.37	pulseaudio
3167	coti	20	0	1308m	352m	48m	S	2,3	4,4	134:44.69	chrome
1734	root	20	0	260m	153m	27m	S	1,7	1,9	185:30.21	Xorg
27381	coti	20	0	993m	127m	28m	S	1,3	1,6	1:26.38	chrome
29027	coti	20	0	966m	95m	22m	S	1,0	1,2	0:04.14	chrome
22163	coti	20	0	231m	46m	14m	S	0,7	0,6	0:14.43	emacs
27744	coti	20	0	945m	124m	20m	S	0,7	1,6	0:01.98	chrome
1788	avahi	20	0	35928	3700	1488	S	0,3	0,0	23:25.46	avahi-daemon
4301	coti	20	0	386m	24m	12m	S	0,3	0,3	0:55.60	gnome-terminal

Commande free : mémoire utilisée

- En ko par défaut, sinon octets, Mo, Go, compatible avec les humains :
options -b, -m, -g, -h

```
coti@maximum:~$ free -h
              total        used         free       shared    buffers         cached
Mem:           7,8G         7,0G         827M          0B          192M         2,7G
-/+ buffers/cache: 4,1G         3,7G
Swap:          7,4G         2,6M         7,4G
```

Utilisation de la mémoire : commande vmstat

- Possibilité de mesurer au cours du temps

```
coti@maximum:~$ vmstat 2 5
procs -----memory----- --swap-- ----io---- -system-- ----cpu-----
 r b  swpd  free  buff  cache  si  so   bi  bo   in  cs us sy id wa
0 1   2644 802256 196804 2802480  0  0   0  0   1   4   2  1  0 99  0
0 0   2644 789608 196804 2803268  0  0   0  0  6362 9583  5  2  91  2
0 0   2644 790072 196804 2803332  0  0   0  0  3893 6205  3  1  94  2
0 1   2644 774516 196804 2803684  0  0   0  0  8501 12415  6  2  87  5
1 0   2644 770200 196804 2803800  0  0   0  0  3082 5239  3  1  85  11
```

Communications inter-processus : IPC V5, listées avec la commande ipcs

```

coti@maximum:~$ ipcs
----- Segment de mémoire partagée -----
clé      shmuid      propriétaire perms      octets      nattch      états
0x00000000 0          coti        600        393216      2           dest
0x00000000 32769     coti        600        393216      2           dest
0x00000000 342130690 coti        777        2050328     2           dest
0x00000000 273055747 coti        777        561600      2           dest
[...]
----- Tableaux de sémaphores -----
clé      semid      propriétaire perms      nsems
----- Queues de messages -----
clé      msqid      propriétaire perms      octets utilisés messages

```

Fichiers ouverts : lsof

- Sous Unix tout est fichier : sockets IP avec l'option -i

```
coti@maximum:~$ lsof -i
```

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OFF	NODE	NAME
icedove-b	3160	coti	56u	IPv4	1816971	0t0	TCP	maximum:45516->wg-in-f109.1e100.net:imaps (ESTABLISHED)
icedove-b	3160	coti	58u	IPv4	18438	0t0	TCP	maximum:55337->mail.iutv.univ-paris13.fr:imaps (ESTABLISHED)
icedove-b	3160	coti	59u	IPv4	2022424	0t0	TCP	maximum:56597->mail:imaps (ESTABLISHED)
icedove-b	3160	coti	69u	IPv4	2022443	0t0	TCP	maximum:32872->mail.iutv.univ-paris13.fr:imaps (ESTABLISHED)
icedove-b	3160	coti	76u	IPv4	2496423	0t0	TCP	maximum:32953->par08s09-in-f19.1e100.net:http (ESTABLISHED)
icedove-b	3160	coti	79u	IPv4	1162259	0t0	TCP	maximum:59881->mail:imaps (ESTABLISHED)

- Passage du chemin vers un répertoire : fichiers se trouvant dans ce répertoire et ses sous-répertoires

```
coti@maximum:~$ lsof /
```

COMMAND	PID	USER	FD	TYPE	DEVICE	SIZE/OFF	NODE	NAME
bash	1746	coti	rtd	DIR	8,1	4096	2 /	
bash	1746	coti	txt	REG	8,1	975488	5373975	/bin/bash
bash	1746	coti	mem	REG	8,1	47616	4769217	/lib/x86_64-linux-gnu/libnss_files-2.13.so
bash	1746	coti	mem	REG	8,1	43552	4769213	/lib/x86_64-linux-gnu/libnss_nis-2.13.so
bash	1746	coti	mem	REG	8,1	89056	4769211	/lib/x86_64-linux-gnu/libnsl-2.13.so

Sockets ouvertes : commande netstat

- Possibilité de préciser le protocole : -u pour UDP, -t pour TCP
- Donne le PID et le nom du programme utilisant la socket
- Port local, adresse et port distants
- État de la socket

```
coti@maximum:~$ netstat -lapute
```

```
Connexions Internet actives (serveurs et établies)
```

Proto	Recv-Q	Send-Q	Adresse locale	Adresse distante	Etat	User	Inode	PID
tcp	0	0	*:sunrpc	*:*	LISTEN	root	3906	-
tcp	0	0	*:ssh	*:*	LISTEN	root	7730	-
tcp	0	0	localhost:ipp	*:*	LISTEN	root	1861089	-
tcp	0	0	localhost:smtp	*:*	LISTEN	root	6142	-
tcp	0	0	*:841	*:*	LISTEN	root	10374	-
tcp	0	0	*:33388	*:*	LISTEN	root	10711	-
tcp	0	0	*:42989	*:*	LISTEN	statd	3925	-
tcp	0	0	maximum:41813	wg-in-f125.1e100.n:5223	ESTABLISHED	coti	1874088	274
tcp	0	0	maximum:54638	magi.univ-paris13.:2822	ESTABLISHED	coti	2025047	316
tcp	0	0	maximum:36752	we-in-f109.1e100.:imap	ESTABLISHED	coti	1872420	316

État des cartes réseau : commande ifconfig

- Utilisé par le super-utilisateur pour configurer la carte réseau
- En lecture seule pour les autres utilisateurs
- Donne notamment le nombre de paquets erronés reçus

```
coti@maximum:~$ /sbin/ifconfig
eth0      Link encap:Ethernet HWaddr d4:be:d9:9c:7a:9c
          inet adr:10.10.0.217 Bcast:10.10.255.255 Masque:255.255.0.0
          adr inet6: fe80::d6be:d9ff:fe9c:7a9c/64 Scope:Lien
          UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
          RX packets:42594748 errors:0 dropped:0 overruns:0 frame:0
          TX packets:28853636 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 lg file transmission:1000
          RX bytes:18842740245 (17.5 GiB) TX bytes:26078786297 (24.2 GiB)
          Interruption:20 Mémoire:e4c00000-e4c20000
```

Espace restant sur les disques : df (disk free)

```
coti@maximum:~$ df
Sys. fich.                1K-blocks    Util. Disponible  Uti% Monté sur
/dev/sda1                 96120588    12105232    79132620    14% /
tmpfs                    1630668     92104      1538564     6% /tmp
/dev/sdb5                 384499764   203560     364764684   1% /home
lipn-sfa:/export4/vol04/coti 1922471424 1629569536 195246080   90% /users/coti
```

Espace utilisé par un fichier : du (disk used)

```
coti@maximum:~$ sudo du -sh /
```

Attention : méfiance si du et df ne donnent pas un résultat cohérent...

Statistiques d'entrée-sorties sur les disques : iostat

```
coti@maximum:~$ iostat
Linux 3.2.0-3-amd64 (maximum) 27/09/2012 _x86_64_ (8 CPU)

avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           0,71    0,00    0,14    0,22    0,00   98,93

Device:            tps    kB_read/s    kB_wrtn/s    kB_read    kB_wrtn
sda                 0,33         1,19         5,45     1463193     6703576
sdb                 0,00         0,01         0,00         6456         120
```

Les constructeurs placent souvent des **sondes de températures** à des endroits critiques : CPU, disques, GPU...

- Sous Linux : utilitaire `lm_sensors`

```
coti@maximum:~$ sensors
coretemp-isa-0000
Adapter: ISA adapter
Physical id 0:  +34.0 C  (high = +80.0 C, crit = +98.0 C)
Core 0:         +30.0 C  (high = +80.0 C, crit = +98.0 C)
Core 1:         +31.0 C  (high = +80.0 C, crit = +98.0 C)
Core 2:         +32.0 C  (high = +80.0 C, crit = +98.0 C)
Core 3:         +32.0 C  (high = +80.0 C, crit = +98.0 C)
```

Pseudo-système de fichiers /proc

/proc est un **pseudo-système de fichiers** : on y accède (en lecture seule) comme à un système de fichiers monté comme un système de fichiers, mais ça n'en est pas un.

- Utilisé par le noyau pour représenter des informations sur les processus qui tournent
- /proc/<pid> : répertoire relatif au processus de pid <pid>

Contenu :

```
coti@maximum:~$ sudo ls /proc/772
[sudo] password for coti:
attr      coredump_filter  io mountstats    pagemap      stat
autogroup cpuset          limits net        personality  statm
auxv      cwd             loginuid ns           root         status
cgroup    environ        maps numa_maps    sched        syscall
clear_refs exe             mem oom_adj      sessionid    task
cmdline   fd             mountinfo oom_score    smaps        wchan
comm      fdinfo         mounts oom_score_adj  stack
```

- `cmdline` : ligne de commande ayant lancé le processus
- `statm` : statistiques sur la mémoire (taille, pages partagées...)
- `fd` : sous-répertoire contenant des liens vers tous les fichiers ouverts par le programme

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées**
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing

Tâches répétitives

But : effectuer des **tâches répétitives, planifiées** :

- Sauvegarde de fichiers toutes les heures, mise à jour du système toutes les sections, indexation de fichiers...

Sous Unix : outil `crontab`

- Définition dans des tables de ce qui est répété et à quel moment
- Par utilisateur ou pour le système

Utilisateurs autorisés/interdits : définis dans les fichiers `/etc/cron.allow` ou `/etc/cron.deny`

- Visualisation de la table d'un utilisateur : `crontab -l`
- Édition de la table d'un utilisateur : `crontab -e`
- Pour tout le système : fichier `/etc/crontab`

Syntaxe de la table

Tableau contenant 6 ou 7 champs :

- Le moment auquel la tâche doit être exécutée (5 champs)
- Pour le crontab du système : l'identité sous laquelle on l'exécute
- La tâche à exécuter

Définition du moment auquel exécuter la tâche :

- 5 champs : minute, heure, jour du mois, mois, jour de la semaine
- on donne la valeur numériquement ou en 3 lettres pour les jours et les mois
- * si c'est à chaque fois, possibilité de donner une liste ou un intervalle

Exemple de ligne du crontab utilisateur :

- Minute 0, heure 7, tous les mois, du lundi au vendredi :

```
0 7 * * 1-5    echo "Debout feignasse !"
```

→ s'exécutera tous les jours 1 à 5 (lundi au vendredi) à 7 :00

Exemple de ligne du crontab système :

- Toutes les minutes, toutes les heures, tous les jours :

```
0 0 * * *    root    updatedb
```

→ s'exécutera toutes les minutes sous l'identité du super-utilisateur

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation**
- 10 Introduction à la virtualisation
- 11 Cloud computing

Qu'est-ce que le système d'exploitation ?

Ordinateur = ensemble de ressources matérielles

- Disque dur, barrettes mémoire, microprocesseur(s), périphériques...

Problème : comment les utiliser ? Comment y accéder ?

Première possibilité : y accéder directement

- Communiquer directement sur le bus système, commander le matériel des périphériques
- Complexe !!

Autre possibilité

- Fournir un système qui propose une **abstraction du matériel** et offre une **interface simple** aux applications pour y accéder
- **Système d'exploitation de l'ordinateur**

Système d'exploitation : exemple des disques

Variété des systèmes de stockage

- **Diversité** des types d'interfaces matérielles (IDE, SATA, SATA-II, USB, SCSI...)
- **Diversité** des systèmes de fichiers sur ces disques (ext2/ext3/ext4, ReiserFS, Btrfs...)

Du point de vue de l'application :

- Interface **unique** !
- Fonctions **simples** comme ouvrir un fichier, écrire dedans, se déplacer dans ce fichier, le fermer...
- *Les mêmes fonctions* quel que soit le matériel et le système de fichiers utilisés

Comment réaliser cela ?

→ **Système d'exploitation de l'ordinateur**

Système multi-tâches : plusieurs programmes s'exécutent en même temps

- Plus de programmes que de cœurs disponibles sur la machine !

Besoin d'arbitrer l'accès aux cœurs de la machine

- Un processus a accès à un cœur, les autres attendent (dorment)
- Au bout d'un certain temps, ce processus est mise en attente et un autre a accès au cœur
- ...

On appelle cette opération **l'ordonnancement** .

Ordonnancement des processus sur les cœurs disponibles ?

→ **Système d'exploitation de l'ordinateur**

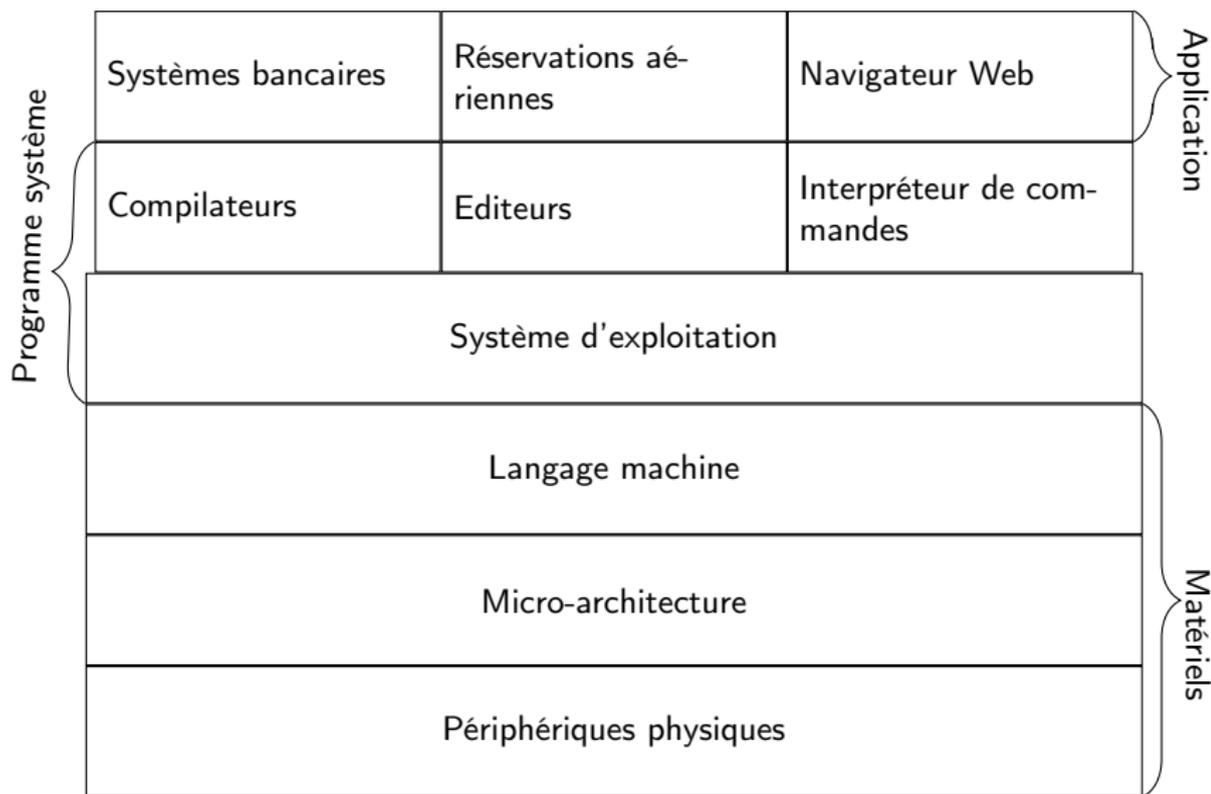
```
coti@maximum:~$ ps aux
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.0	10648	720	?	Ss	avril11	0:18	init
root	2	0.0	0.0	0	0	?	S	avril11	0:01	[kth
root	3	0.0	0.0	0	0	?	S	avril11	0:36	[kso
root	6	0.0	0.0	0	0	?	S	avril11	0:02	[mig
root	7	0.0	0.0	0	0	?	S	avril11	0:08	[wat
root	8	0.0	0.0	0	0	?	S	avril11	0:00	
[migration/1]										
[...]										
coti	4068	0.0	0.0	14264	5016	tty1	S+	avril11	0:00	-bas
coti	4484	1.8	7.4	2001652	608644	?	Rl	avril11	1036:01	ice
[...]										

Définition

Le rôle du système d'exploitation est d'*orchestrer l'accès aux ressources matérielles* des programmes d'exécutant sur la machine. Notamment, il gère l'accès à la mémoire, l'exécution des processus, l'accès aux périphériques via des drivers...

Architecture logicielle du système



Tanenbaum : Systèmes d'Exploitation

Le **noyau** est situé *au cœur* du système d'exploitation

- Il effectue les **tâches fondamentales** : gestion de la mémoire, ordonnancement
- Quelques autres fonctionnalités pour des raisons de performances : piles protocolaires réseaux, systèmes de fichiers

Exemple :

- Allocation mémoire dans un programme C : `malloc`
- Fonction du noyau
- Le noyau conserve une vision globale de la mémoire, ce qui a été alloué et à qui
- Il alloue de la mémoire au processus qui lui a demandé
- Un processus n'a pas le droit d'accéder à de la mémoire qui ne lui appartient pas (segmentation fault)

Espace noyau vs espace utilisateur

Le noyau a accès à **toutes les ressources** matérielles de l'ordinateur

- Toute la mémoire, périphériques...

Les programmes n'ont accès qu'à ce dont ils ont besoin

- en cas de besoin : appels au noyau (`malloc...`) ou aux pilotes matériels

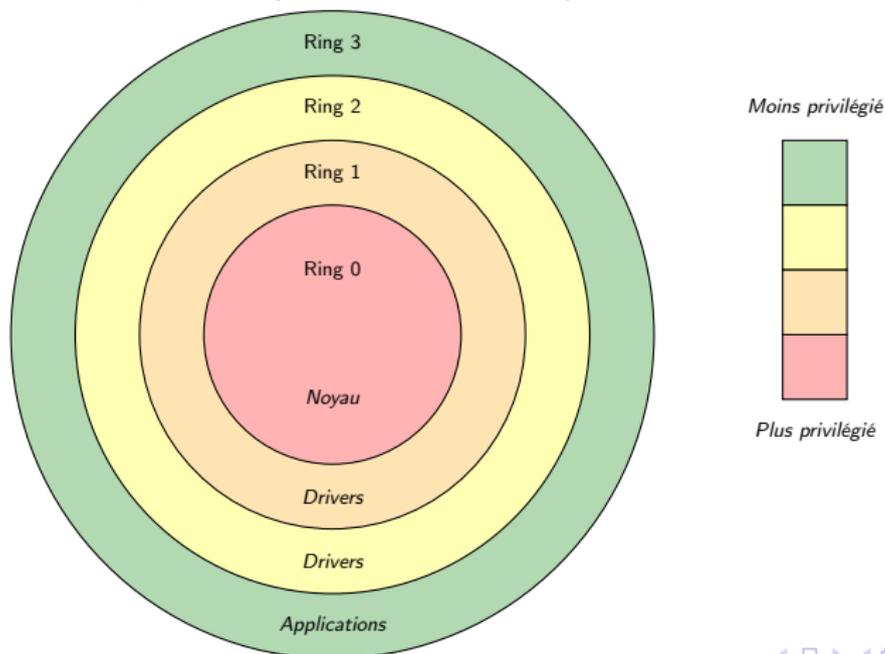
Concept de protection :

- Le noyau s'exécute **en espace noyau**
- Les applications s'exécutent **en espace utilisateur**

Rings de protection

Espace noyau, espace utilisateur

- En réalité : 4 **niveaux de protection** : les **anneaux de protection** (protection rings)
- Espace noyau = ring 0 ; espace utilisateur = ring 3
- Rings 1 et 2 (rings intermédiaires) : pilotes



Le noyau du système

Le caractère central du noyau et son rôle de “chef d'orchestre” ont deux conséquences fondamentales :

- Le noyau est le **premier programme lancé** au démarrage de la machine
- **Un seul noyau** est exécuté en espace noyau

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation**
 - Pourquoi virtualiser
 - Confinement
 - Noyau en espace utilisateur
 - Paravirtualisation
- 11 Cloud computing

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation

Pourquoi virtualiser

Pourquoi utiliser des machines virtuelles ?

- **Exploitation des ressources matérielles**

- Machines puissantes
- Plusieurs serveurs par machine

- **Isolation des services**

- Un service ↔ une machine virtuelles
- En cas de corruption : confinement, non-contamination

- **Migration et sauvegarde**

- En cas de problème : migration de la machine avant la panne
- Sauvegarde des machines virtuelles, redémarrage autre part

Système invité / système hôte

Principe de base : **un seul système en espace noyau** : c'est le **système hôte** .

- Orchestre l'accès aux ressources de la machine

Systèmes virtualisés exécutés sur la machine : **systèmes invités**

- Tournent en **espace utilisateur**
- Considérés par l'ordonnanceur comme des processus comme les autres

Types de systèmes de virtualisation

Émulation : on fait passer une architecture matérielle pour une autre

- Exemple : exécuter un système pour SPARC sur un x86

Principe :

- On a un binaire exécutant un certain jeu d'instructions
- On a une architecture disposant d'un autre jeu d'instructions

L'émulateur **traduit** les instructions du binaire vers celles de l'architecture cible

Virtualisation : on exécute plusieurs machines sur une machine physique hôte.

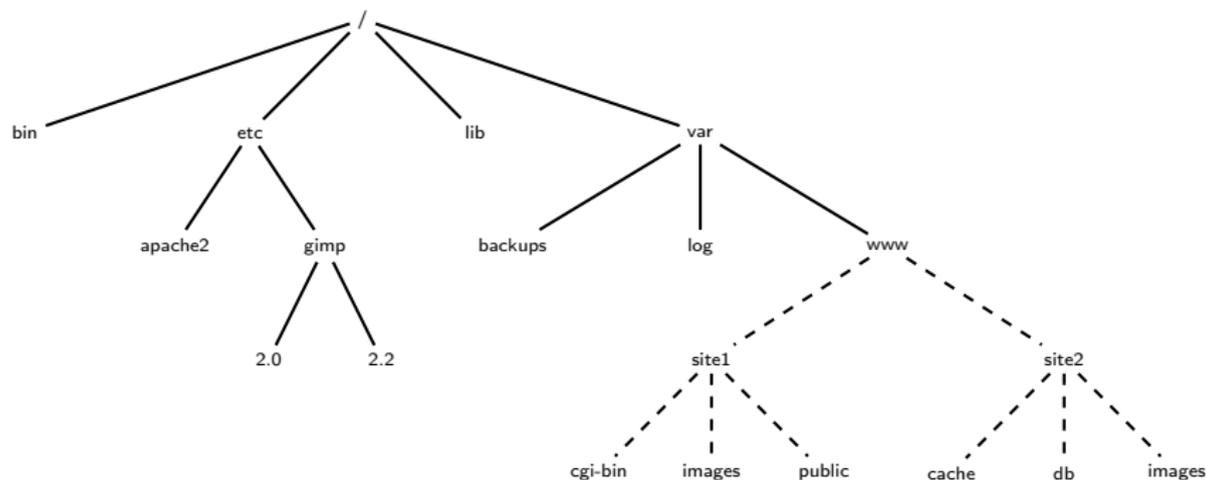
Quatre grandes familles d'outils de virtualisation :

- Les **isolateurs**
- Les **noyaux en espace utilisateur**
- Les **machines virtuelles**
- Les **para-virtualiseurs**

Confinement disque

Outil chroot (change root)

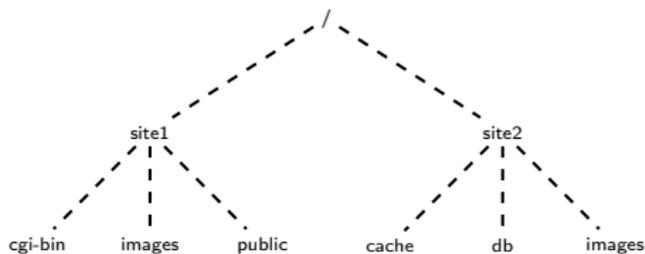
- Pour un processus, on déplace ce qu'il voit comme la racine de l'arborescence de fichiers



Confinement disque

Outil `chroot` (change root)

- Pour un processus, on déplace ce qu'il voit comme la racine de l'arborescence de fichiers



Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation

Confinement disque

On exécute le programme `/bin/bash` dans un espace confiné à `/var/www` :

```
coti@maximum:~$ pwd
/users/coti
coti@maximum:~$ sudo chroot /var/www /bin/bash
bash-4.2# pwd
/
bash-4.2# cd ..
bash-4.2# pwd
/
bash-4.2#
```

Intérêt : le processus exécuté n'a **accès qu'à une sous-arborescence de fichiers**

- Serveurs HTTP, FTP...

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation

Noyau en espace utilisateur

Principe : exécuter un noyau comme n'importe quel programme

- Outil : **User-Mode Linux** (UML)

Le noyau est compilé de façon particulière (options de compilation)

- Se lance comme un exécutable
- Prend en paramètres les systèmes de fichiers fournis

```
root@maximum:/tempo# ./linux-3.11.0-uml ubda=disk.img ubdb=swap.img \  
    root=/dev/ubda mem=256M
```

Avantages :

- Simple, intégré au noyau Linux, bonne isolation

Inconvénients :

- Performances médiocres (conçu pour du débogage)

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation

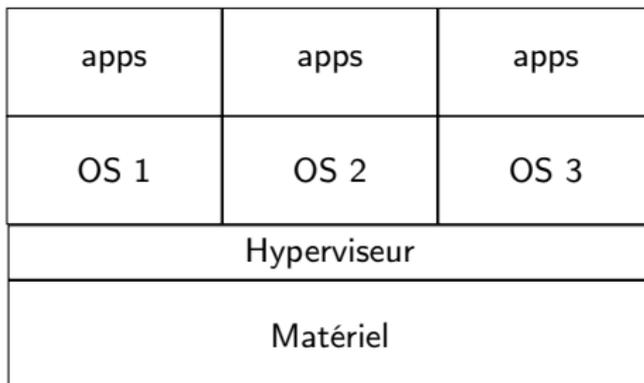
Paravirtualisation

Noyaux invités contrôlés par un **hyperviseur**

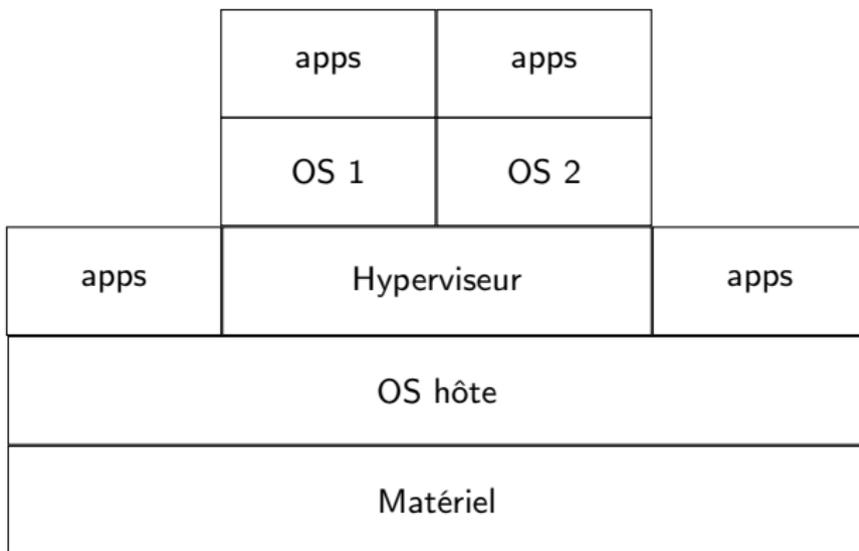
Deux types d'hyperviseurs :

- Hyperviseur de **type I** , ou natif, ou bare metal
 - Exécuté directement sur le matériel de la machine
 - Fine couche logicielle située entre le matériel et le système
 - Tous les noyaux s'exécutant dessus sont des noyaux invités.
 - Exemples : Xen, KVM (intégré au noyau Linux)
- Hyperviseur de **type II** , ou hébergé
 - Se situe entre le système d'exploitation hôte et les systèmes invités
 - Exemples : VirtualBox, QEMU et VMware Workstation

Hyperviseur de type I



Hyperviseur de type II



Noyaux exécutés par l'hyperviseur

Les noyaux exécutés par l'hyperviseur tournent en **espace utilisateur**

- Soit dans le ring 3
- Soit dans les rings intermédiaires (1 ou 2)

Les noyaux invités n'ont donc **pas accès à toutes les ressources** matérielles

- Certains appels doivent être remplacés par des appels à des fonctions de l'hyperviseur
- Les systèmes invités doivent donc être modifiés

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 **Cloud computing**
 - Principes du cloud computing
 - Modèle économique
 - Types de cloud

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation

Principe du cloud computing

Principe de base : l' **externalisation**

- Les ressources ne sont pas chez soi (dans une entreprise, chez un particulier...) mais chez un **prestataire de service**

Différents types de cloud :

- Cloud de stockage : Dropbox, Google Drive...
- Cloud applicatif : Google Docs, SlapOS...

Caractéristiques :

- Accès via le réseau (Internet)
- Transparence : localisation, mobilité, accès
- Généralement, accès via le navigateur Web
- Souplesse! Externalisé = on loue des ressources, modifications rapides (augmentation de la capacité, etc)

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation

Modèle économique

On loue des ressources, un service

- Définition de façon contractuelle des ressources : capacité, nombre de cœurs, fiabilité...

Définition d'un **Service Level Agreement** (SLA)

- Si le SLA n'est pas respecté : pénalités pour le prestataire de service

Plan du cours

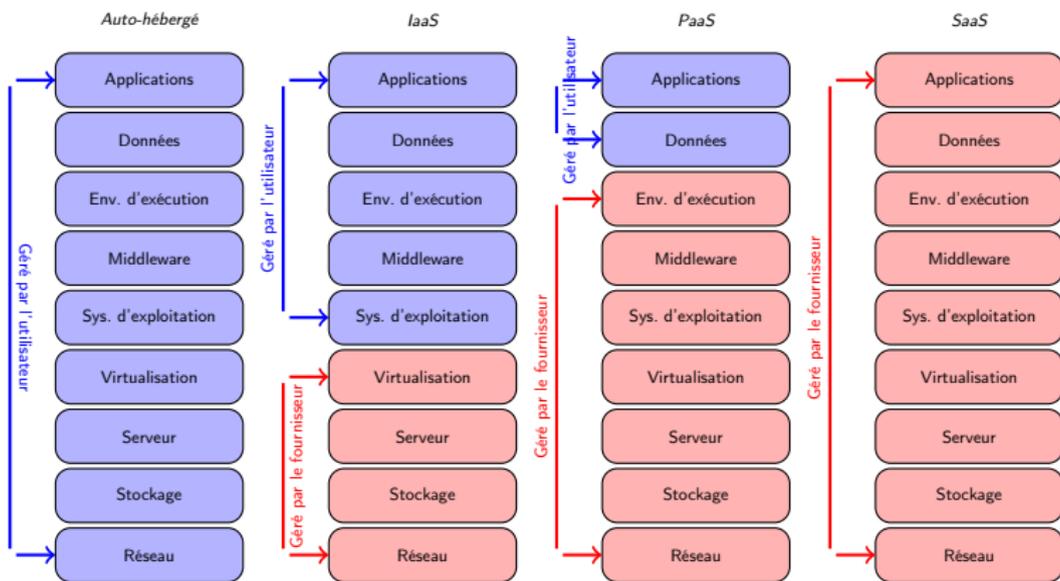
- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation

Types de cloud

Le niveau de service dépend du type de cloud.

- Cas extrême (non cloud) : **machine auto-hébergée** . L'utilisateur s'occupe de tout.
- **Infrastructure as a Service** : le prestataire fournit l'*infrastructure matérielle*, le client exécute son propre système sur un système de virtualisation. Exemple : Amazon EC2
- **Platform as a Service** : le prestataire fournit l'*environnement d'exécution*, le client exécute son application et stocke ses données sur le cloud. Exemple : hébergement Web.
- **Service as a Service** : le prestataire fournit un *service* préinstallé : blog, application... Le client ne fait que l'exécuter. Exemple : Google Docs, interface Gmail...

Types de cloud



Évolution

Terminaux de plus en plus légers : tablettes, smartphones...

- Moins puissants, peu de capacités de stockage
- Accès au réseau !

Services performants, latence réseau faible, connexions permanentes

- Utilisation de plus en plus important de services distants
- Externalisation des applications, du stockage

On se dirige de plus en plus vers des **clients légers** et des applications distantes, centralisées

- Retour aux mainframes !
 - Le terminal ne fait que de l'affichage

Plan du cours

- 1 Introduction du module
- 2 Administration système
- 3 Système multi-utilisateurs
- 4 Limites fixées aux utilisateurs
- 5 Arborescence de fichiers
- 6 Stockage sûr
- 7 Métrologie
- 8 Tâches planifiées
- 9 Système d'exploitation
- 10 Introduction à la virtualisation
- 11 Cloud computing